UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería de Servicios y Sistemas de Telecomunicación

TRABAJO FIN DE GRADO

RE-IDENTIFICACIÓN DE PERSONAS Y VEHÍCULOS

Carlos González Ruiz
Tutor: Álvaro García Martín
Ponente: Álvaro García Martín

Julio 2020

RE-IDENTIFICACIÓN DE PERSONAS Y VEHÍCULOS

Carlos González Ruiz
Tutor: Álvaro García Martín
Ponente: Álvaro García Martín

Video Processing and Understanding Lab
Dpto. 111
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Julio 2020

Trabajo parcialmente financiado por el Gobierno de España bajo el proyecto TEC2017-88169-R (MobiNetVideo)

Resumen

La principal atención de este trabajo de fin de grado es el estudio de sistemas de re-identificación de personas y vehículos basados en la combinación de características profundas y características tradicionales que describen los datos utilizados.

En primer lugar se estudiará el estado del arte para comprender las técnicas de *deep learning* basadas en redes neuronales que serán imprescindibles para extraer las características necesarias en tareas de re-identificación.

En segundo lugar, se describirán los conjuntos de datos utilizados y sus respectivos atributos tanto para el de personas como el de vehículos. Además, se incluirá una pequeña introducción al entorno de desarrollo utilizado (Pytorch) y a los modelos preentrenados escogidos para realizar los experimentos. Una vez explicados los anteriores conceptos, se hará un resumen del diseño implementado a partir de ellos.

Por último, se explicarán las métricas utilizadas para evaluar y extraer resultados de re-identificación así como diferentes técnicas para mejorar dichos resultados. Se realizarán numerosos experimentos empleando varios modelos entrenados con distintos parámetros para extraer conclusiones generales de los resultados y posteriormente realizar la combinación de las características para observar el efecto en el resultado final.

Antes de finalizar el trabajo de fin de grado se valorarán los resultados obtenidos y se proporcionarán pruebas visuales que demuestren el efecto conseguido y para acabar se propondrán nuevas vías de investigación que se podrían realizar en un futuro.

Palabras clave

Aprendizaje profundo, transferencia de aprendizaje, re-clasificación, características de emparejamiento acumulativo (CMC), media de la precisión promedio (mAP).

Abstract

The main focus of this final degree project is the study of people and vehicle re-identification systems based on the combination of deep learning characteristics and traditional characteristics that describe the data used.

First, the state of the art will be studied to understand the deep learning techniques based on neural networks that will be essential to extract the necessary characteristics in re-identification tasks.

Secondly, the data sets used and their respective attributes for both people and vehicles will be described. In addition, a short introduction to the development environment used (Pytorch) and the pre-trained models chosen to carry out the experiments will be included. Once the previous concepts have been explained, a summary of the implemented design will be made from them.

Finally, the metrics used to evaluate and extract re-identification results will be explained, as well as different techniques to improve said results. Numerous experiments will be carried out using several trained models with different parameters to draw general conclusions from the results and then perform the combination of the characteristics to observe the effect on the final result

Before finishing the final project, the results obtained will be evaluated and visual tests will be provided to demonstrate the effect achieved, and to finish, new research routes will be proposed that could be carried out in the future.

Keywords

Deep learning, transfer learning, re-ranking, cumulative matching characteristics (CMC), mean Average Precision (mAP)

Agradecimientos

En primer lugar quiero agradecer a mi tutor Álvaro García por darme la oportunidad de realizar este trabajo de fin de grado y por guiarme de manera efectiva en la realización del mismo. También quería agradecer tanto a Álvaro, como a todo el equipo del VPU por su ayuda ante las complicaciones que han ido surgiendo.

También dar las gracias a mis padres y a mi hermana por su confianza y apoyo constante durante todos estos años que han sido claves para la realización de mis objetivos personales y profesionales.

Agradecer también a todas mis amigos y compañeros de carrera que han sido un pilar fundamental durante la carrera. Ha sido una suerte poder conocerlos y haber formando una gran piña ya que gracias a ellos mi paso por la universidad ha sido más sencillo.

Por último, agradecer a mis amigos más cercanos por todos esos buenos momentos que me han permitido desconectar cuando más lo necesitaba.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Motivación

El aprendizaje profundo (*deep learning*) se ha convertido en uno de los áreas más populares dentro del campo de la inteligencia artificial (*artificial intelligence*) debido a sus infinitas aplicaciones como la clasificación de imágenes. La motivación de este trabajo de fin de grado es poder investigar en este tipo de aplicaciones y técnicas de *deep learning* basadas en redes neuronales convolucionales para adentrarse en el campo de la re-identificación de imágenes.

Este proyecto se centra en la re-identificación de personas [1] y vehículos los cuales están experimentando un enorme crecimiento debido a su alta demanda en la actualidad y al notable aumento de cámaras intaladas en edificios y a lo largo de la vía pública. La tarea principal es la de reconocer a una persona o vehículo que han sido previamente captadas por una serie de cámaras en un entorno determinado. La investigación en este tipo de tecnología puede proporcionar un gran número de herramientas útiles en el campo de la seguridad. Concretamente, es bastante útil en sistemas de video vigilancia inteligente que es un área importante de investigación por ser clave en la lucha contra el crimen y el terrorismo.

Gracias a la re-ídentificación es posible realizar un seguimiento de las personas y vehículos para permitir un análisis de sus actividades y comportamientos. Posibles aplicaciones de estos sistemas pueden ser el seguimiento de las personas en determinados entornos por motivos de seguridad como puede ser en un aeropuerto para prevenir amenazas o simplemente mantener un control monitorizado. En el caso de la re-identificación de vehículos se puede realizar el mismo tipo seguimiento para detectar anomalías en el tráfico que puedan evitar algún tipo de accidente.

Aunque las principales ventajas de los sistemas de re-identificación estén enfocadas a mejorar la seguridad de las ciudades, también es posible mejorar su inteligencia ya que a partir de los datos de re-identificación se puede conseguir una mejor optimización de la vía pública y por ejemplo mejorar con ello la eficiencia del tráfico

Para conseguir todo este tipo de aplicaciones y objetivos es necesario disponer de un buen sistema de re-identificación de imágenes y por eso en este trabajo además de probar y analizar varios modelos de redes neuronales que aprendan sobre las identidades

de las personas y vehículos se propone emplear características de los mismos, como la ropa en personas o la marca en vehículos con el objetivo de mejorar los resultados del sistema de re-identificación.

1.2. Objetivos

El objetivo de este trabajo de fin de grado es la evaluación y comparación de diferentes características para la descripción e re identificación de personas y vehículos en escenarios de video seguridad. La finalidad es el estudio de la combinación de característcas de aprendizaje profundo con características extraídas a partir de atributos o metadatos para observar si se consigue mejorar mucho, poco o nada los resultados de re-identificación. Para llegar al objetivo final se realizarán una serie de objetivos intermedios:

- Se realizará un estudio previo sobre los elementos fundamentales del *deep learning* y en concreto sobre las redes neuronales convolucionales que serán las encargadas de generar las caracteríticas necesarias para el sistema de re-identificación
- A partir del estudio realizado se tomarán algunas variaciones de los hiperparámetros más importantes para generar resultados de re-identificación y observar cuales tienen mejor desempeño
- Por otro lado se entrenará un clasificador multiclase con los metadatos o atributos de los conjuntos de datos para posteriormente extraer tanto la probabilidad que tiene cada imagen de poseer los atributos como el vector de características de cada una de ellas.
- Una vez determinado los mejores parámetros de los modelos se probará a combinar las características de aprendizaje profundo con las características extras (atributos) proporcionadas por el clasificador multiclase
- Tras diversas formas de combinación se analizará los resultados obtenidos para observar si se ha conseguido mejorar los resultados iniciales o por el contrario las pruebas han sido fallidas.
- Por último se proporcionarán varias pruebas visuales de ambos conjuntos de datos que traten de demostrar el efecto conseguido durante el proyecto.

1.3. Organización de la memoria

Esta memoria consta de los siguientes capítulos:

- Capítulo 1 Descripción de los objetivos y estructura del trabajo.
- Capítulo 2 Análisis del estado del arte y estudio de los conceptos más importantes necesarios para el desarrollo del trabajo .
- Capítulo 3 Presentación de los conjuntos de datos utilizados durante el trabajo así como el diseño y desarrollo del entrenamiento de los modelos.

- Capítulo 4 Estudio y análisis de las medidas y algoritmos necesarios para la evaluación de los sistemas de re-identificación y realización de los experimentos.
- Capítulo 5 Valoración de los resultados obtenidos en su conjunto y propuestas de nuevas vías de investigación.

Capítulo 2

Estado del arte

2.1. Introducción

El Aprendizaje automático (*Machine learning*) es un concepto que ha experimentado un enorme crecimiento en los últimos años debido a sus infinitas aportaciones en inteligencia artificial. Realmente pertenece al campo de la inteligencia artificial y su objetivo es dotar a las máquinas de la capacidad de aprender a partir de un gran conjunto de datos, con la idea principal de conseguir realizar tareas sin haberlas programado específicamente.

El aprendizaje puede ser de 3 tipos:

- Aprendizaje Supervisado (*Supervised learning*): Esta técnica emplea un conjunto de datos de entrada etiquetados que les sirven a las máquinas como ejemplos para aprender sobre ellos y ser capaces de obtener unos buenos resultados ante unos datos de entrada similares.

- Aprendizaje no supervisado (*Unsupervised learning*): La principal diferencia que presenta este tipo de aprendizaje es que usa como datos de entrada un conjuntos de datos que no se encuentran clasificados o etiquetados ya que trata de descubrir patrones y características de dichos datos para agruparlos y estructurarlos.

- Aprendizaje por refuerzo (*Reinforcement learning*): El objetivo de este tipo de aprendizaje es encontrar el mejor modelo o comportamiento para llegar a maximizar el resultado en el entorno que se realiza y para ello se emplea el principio de prueba y error.

Según el tipo de aprendizaje y las posibles aplicaciones existen distintas técnicas de *Machine learning* como los modelos de regresión, análisis de grupos (*clusterización*), árboles de decisión... Sin embargo, una de las técnicas más importantes dentro del ámbito del Machine Learning son las redes neuronales (*neural networks*) las cuales tratan de imitar el comportamiento biológico del cerebro humano.

Las redes neuronales son capaces de aprender de manera jerarquizada haciendo uso de distintas capas o niveles y a medida que se atraviesan dichas capas se aprenden desde conceptos muy concretos hasta conceptos más abstractos. Con el paso de los años

las redes neuronales han ido aumentando el número de capas convirtiéndose en algoritmo más complejos pasando a ser conocidos como algoritmos de aprendizaje profundo.

El aprendizaje profundo (*deep learning*) es un subconjunto dentro del campo del *machine learning* construido a partir del principio de las redes neuronales y ofrece una solución al tratamiento de datos masivos ya que permite descubrir o reconocer patrones, características y relaciones de dichos datos indetectables para el ser humano.

Las aportaciones del *deep learning* son muy amplias y actualmente podemos encontrar su aplicación en campos de la medicina, el reconocimiento de voz o en identificación de imágenes, entre otros. Para la identificación/clasificación de imágenes la red neuronal profunda más usada es la red neuronal convolucional o *convolutional neuronal network* (CNN) de la cuál profundizaremos en la sección 2.3.

2.2. Redes Neuronales

Las redes neuronales son un modelo de inteligencia artificial que tratan de imitar el comportamiento biológico del cerebro humano para realizar tareas inalcanzables por algoritmos convencionales. A partir de una serie de entradas, las redes neuronales se encargan de procesar la información y generar relaciones entre los datos con el objetivo de generar unos resultados.

A pesar de haber sido inventadas sobre los años 60 no ha sido hasta hace unos años cuando están siendo realmente aprovechadas debido al crecimiento de la tecnología como es el caso de la capacidad de cómputo sobre todo la potencia de cálculo de las GPUs ya que permiten paralelizar muchos procesos reduciendo el tiempo de ejecución.

2.2.1. Neurona

A la unidad básica de procesamiento dentro de una red neuronal se le conoce como neurona. Cada neurona se encarga de realizar un cálculo interno a partir de una serie de datos de entrada con el objetivo de generar una salida que pueda ser utilizada por otras neuronas dentro de la red neuronal.

Las diferentes partes de la neurona se pueden observar en la Figura 2.1:

- Datos de entrada: Pueden ser tanto los datos iniciales como las salidas de otras neuronas y van asociadas a un peso que determina su importancia dentro de la neurona. Los pesos son parámetros que se ajustan a medida que se entrena la red neuronal.
- Cálculo interno: Realiza la multiplicación de cada entrada por su respectivo peso y posteriormente se suman alcanzando un valor (suma ponderada). A este resultado se le suma otro valor formado por la multiplicación de una variable por un peso asociado que se le conoce como valor de sesgo o bias. Dicha variable esta siempre asignada a '1' por lo que se modifica con su respectivo peso.
- Función de activación: Añade deformaciones no-lineales al cálculo interno para obtener un rango determinado de valores de salida. .

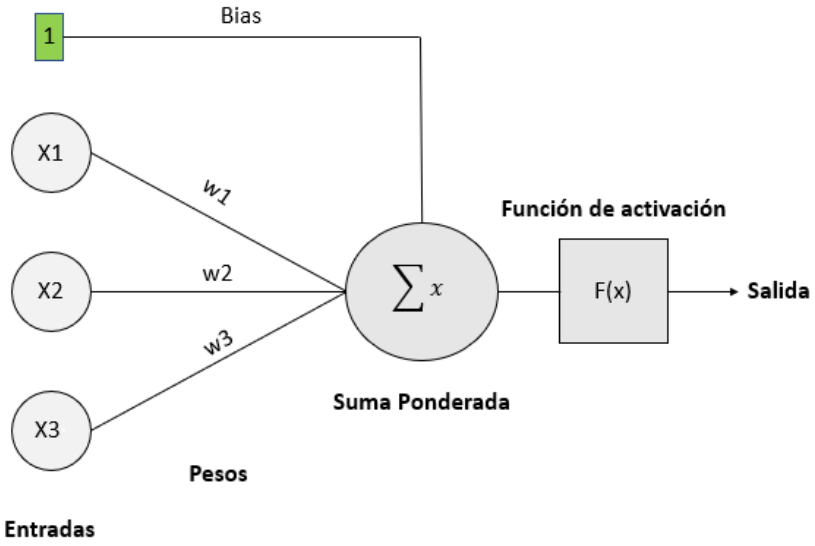

Figura 2.1: Esquema de una neurona

Las funciones de activación mas utilizadas son la función sigmoide y la función RELU. La función sigmoide produce una saturación de los valores grandes a '1' y una saturación de los valores más pequeños a '0' usando la fórmula 2.1 consiguiendo un rango ideal para representar probabilidades.

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{2.1}$$

Por su parte, la función RELU (*Rectified Lineal Unit*) asigna los valores de entrada negativos a '0' y con los positivos se comporta de manera lineal siguiendo la función 2.2:

$$R(z) = max(0, z) \tag{2.2}$$

Debido al buen desempeño que la función RELU presenta cuando los datos de entrada son imágenes es la función de activación más común de las redes neuronales convolucionales (ver sección 2.3)

2.2.2. Estructura

Una sola neurona no tiene la capacidad de modelar muchos datos por sí sola, por eso las neuronas se organizan en niveles o capas interconectadas formando en su conjunto la red neuronal.

La estructura de las redes neuronales presenta 3 tipos de capas (Figura 2.2):

- Capa de entrada (*Input layer*): Es la primera capa de una red neuronal ya que se encuentra compuesto por el conjunto de neuronas que reciben los datos que se desean procesar.

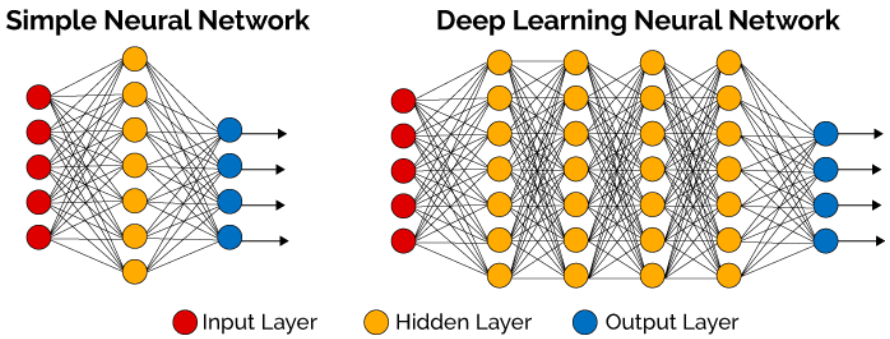

Figura 2.2: Red neuronal simple (izquierda). Red neuronal profunda (derecha). Se observan las 3 capas características de las redes neuronales: entrada, ocultas y salida (Fuente: [2])

- Capas ocultas (*Hidden layers*): Son las capas intermedias de la red neuronal y la componen las neuronas encargadas de procesar y transmitir la información. A mayor número de capas ocultas, mayor profundidad tiene la red neuronal.
- Capa de salida (*Output layer*): Es la última capa que forma una red neuronal y cuyas neuronas aportan los resultados de salida.

La estructura más común de una red neuronal es la red neuronal multicapa que puede tener sus neuronas totalmente o parcialmente conectadas pero según la finalidad y el conjunto de datos de entrada existen otro tipo de arquitecturas como la red neuronal recurrente o *recurrent neural network* (RNN) y la red neuronal convolucional o *convolutional neural network* (CNN). Una red tipo RNN es especialmente recomendada para tratar con datos secuenciales como es el caso de la voz. Por su parte, la CNN es de bastante utilidad cuando se disponen de datos con estructura espacial como pueden ser las imágenes por lo que se profundizará en este tipo de arquitectura (Ver sección 2.3).

2.2.3. Entrenamiento

Para que una red neuronal sea capaz de obtener unos buenos resultados antes debe haber sido entrenada. El entrenamiento permite a las redes neuronales aprender por sí solas que parámetros utilizar a partir de los datos de entrada. Para ello se hace uso de uno de los algoritmos de optimización más usados en aprendizaje automático como es el descenso por gradiente apoyado de un algoritmo llamado propagación hacia atrás o *backpropagation* que es la base del aprendizaje supervisado.

Para empezar, se hace pasar una entrada cualquiera por todas las capas de la red neuronal, que tendrá sus parámetros inicializados aleatoriamente, hasta obtener una salida. Por norma general, el resultado de esta primera pasada estará muy alejado del objetivo de la red neuronal.

A continuación, se hace uso de una función de coste para determinar el error del valor estimado respecto al valor real. A partir de dicho error se inicia el algoritmo de *backpropagation* cuyo funcionamiento básico trata de realizar una propagación del error desde la capa final hasta la capa inicial (hacia atrás) con el objetivo de dar a cada neurona un error según la responsabilidad que haya tenido en el resultado final. Esos

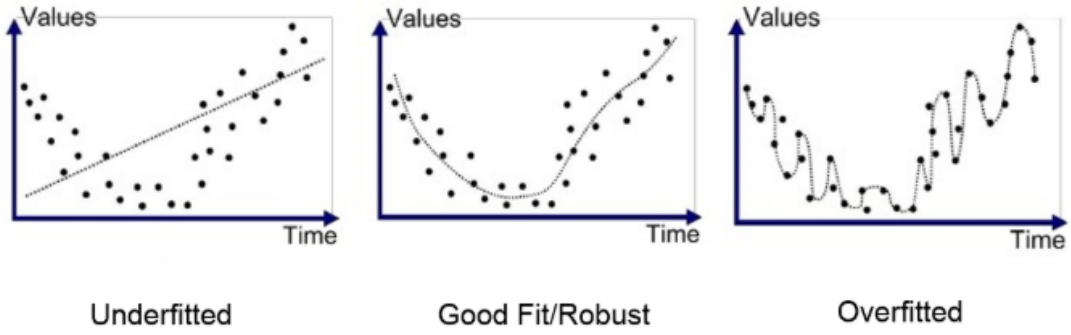

Figura 2.3: Ejemplo de un apredizaje subajustado (underfitted), sobreajustado (overfitted) y bien ajustado (good fit) (Fuente: [4])

errores son utilizados para calcular las derivadas parciales de la función de coste de cada parámetro de la red (pesos y sesgos/bias) y así obtener el vector gradiente para aplicar el algoritmo por descenso de gradiente. El vector gradiente nos indica la dirección hacia donde la pendiente asciende en la función de coste por eso para actualizar los pesos se usa este vector siguiendo la fórmula 2.3:

$$Pn = Pa - \alpha * \nabla f \tag{2.3}$$

donde Pn y Pa corresponde al peso nuevo y al peso anterior respectivamente. Por su parte, α representa el coeficiente de aprendizaje (Ver sección 2.2.5) y ∇f corresponde al gradiente. Una vez actualizados los parámetros se repite el proceso hasta minimizar la función de coste.

En resumen, el entrenamiento permite a las redes neuronales aprender automáticamente a ajustar sus parámetros internos. Para ello se utiliza el algoritmo de *backpropagation* para calcular el vector de gradientes de la red neuronal y así poder aplicar la técnica del descenso del gradiente para optimizar la función de coste.

El objetivo del entrenamiento o del aprendizaje es el de dotar a la red neuronal de la capacidad de generalizar el conocimiento sobre los datos con el objetivo de tener un buen rendimiento cuando la entrada de datos sea desconocida para la red (*good fit*). Esto quiere decir que hay que buscar un equilibrio en el aprendizaje ya que pueden darse dos situaciones donde la red neuronal no consigue generalizar el conocimiento correctamente [3] (Ver figura 2.3):

- Subajuste o *underfitting*: Ocurre cuando el modelo no es lo suficientemente complejo como para detectar con precisión las relaciones y características de los datos ya sea por falta de los mismos o por un aprendizaje erróneo o insuficiente.
- Sobreajuste o *overfitting*: Ocurre cuando el modelo aprende o se ajusta demasiado a los datos de entrenamiento y comienza a aprender sus particularidades y no es capaz de obtener un buen rendimiento con datos de entrada nuevos. Como resultado, un modelo que sufra de *overfitting* sera incapaz de obtener un buen rendimiento con nuevos datos de entrada ya que su aprendizaje ha pasado a ser específico y no generalizado a los datos.

2.2.4. Función de coste

La función de coste mide el rendimiento de la red neuronal ya que determina cuál es el error para cada una de las combinaciones de los parámetros de la red. Lo ideal sería obtener un valor nulo con la función de coste lo que implica que el valor estimado por la red es el valor real buscado.

Una de las funciones de coste mas conocida es el error cuadrático medio (MSE) que por medio de la fórmula (2.4):

$$J_{MSE}(y, y_p) = \frac{1}{N}\sum_{x=1}^{n}(y_p(x) - y(x))^2 \tag{2.4}$$

calcula el error buscado entre la estimación (y_p) y el valor esperado (y). En escenarios de clasificación una de las funciones de coste que mas destaca es la pérdida de entropía cruzada o *cross entropy loss* [5] que calcula el error mediante la ecuación (2.5).

$$J(y, y_p) = -\sum_{x_i} y(x_i) * \lg(y_p(x_i)) \tag{2.5}$$

El error cuadrático medio no castiga lo suficiente las clasificaciones erróneas por eso la entropía cruzada es mas popular en tareas de clasificación, sin embargo para tareas de regresión donde la distancia entre dos valores predecibles es pequeña, el error cuadrático medio es la función de coste ideal.

2.2.5. Tasa de aprendizaje

La tasa o el coeficiente de aprendizaje es un hiperparámetro que define la velocidad con que debe aprender la red neuronal, es decir, define cuánto afecta el gradiente a la hora de actualizar los parámetros de la red en cada iteración (ver ecuación 2.3). A menor tasa de aprendizaje más tiempo se necesitará para llegar a minimizar la función de coste (el error) e incluso puede llegar a no sobrepasar los mínimos locales y estancarse, pero a mayor tasa de aprendizaje cabe la posibilidad de no llegar a minimizar la función de coste e incluso llegar a empeorar los resultados, por eso la elección del coeficiente de aprendizaje es clave en las redes neuronales.

En la figura 2.4 se puede observar que empleando tasas de aprendizaje elevadas, el error durante el entrenamiento no llega a minimizarse e incluso llega a empeorar. Sin embargo, empleando una tasa de aprendizaje baja, el error disminuye muy lentamente a lo largo del tiempo y al usar una tasa correcta para un modelo o conjuntos de datos se obtiene una curva similar a la pintada de color verde.

2.2.6. Batchsize y épocas

El tamaño de batch o *Batchsize* es un hiperparámetro de las redes neuronales que indica el número de muestras (datos de entrenamiento) que se van a propagar por la red neuronal antes de actualizar los parámetros de la red.

Existen tres posibles casos de tamaño de batch:

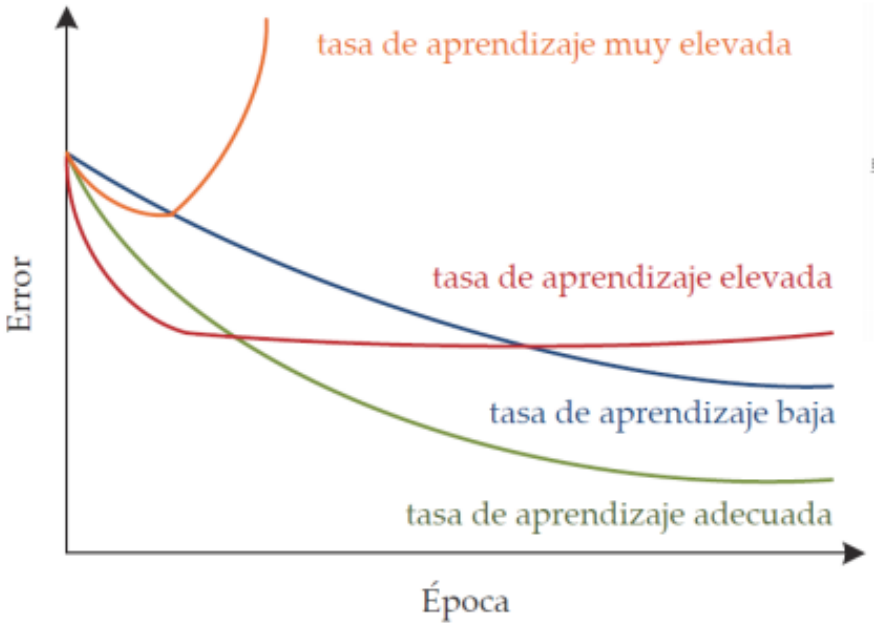

Figura 2.4: Evolución del error sobre el tiempo (épocas) al emplear distintas tasas de aprendizaje. (Fuente: [6])

- Descenso de gradiente por lote (*Batch gradient descent*): Todos los datos de entrenamiento conforman el batch por lo tanto solo existe un batch cuyo tamaño es igual al número de muestras empleadas en el entrenamiento.

- Descenso de gradiente estocástico (*Stochastic gradient descent*): El tamaño de batch es igual a 1 lo que indica que solo se usa una muestra de entrenamiento en cada batch.

- Descenso de gradiente por mini lotes (Mini-batch Gradient descent): El tamaño de batch toma un valor comprendido entre 1 y el tamaño máximo del conjunto de datos.

Usar un tamaño de batch menor requiere de menos memoria y además reduce el tiempo de entrenamiento ya que los parámetros de la red se ven actualizados con cada paso por la red del batch (con su respectivo tamaño de muestras). Por el contrario, emplear un tamaño de batch mayor implica una mayor precisión en la estimación del gradiente lo que puede provocar una convergencia más estable de los resultados.

Por norma general, una sola pasada de todo el conjunto de datos de entrenamiento (divididos o no en batch de distintos tamaños) no son suficiente para entrenar por completo una red neuronal por lo que es necesario emplear varias épocas.

Las épocas (*epochs*) son un hiperparámetro de la red neuronal que indican el número de veces que el conjunto de datos de entrenamiento al completo va a realizar un paso hacia delante y hacia atrás sobre la red (*forward/backward*). No hay un número de épocas que funcione correctamente siempre sino que depende de la tarea de la red, el conjunto de datos empleados y la profundidad de aprendizaje buscada, entre otros.

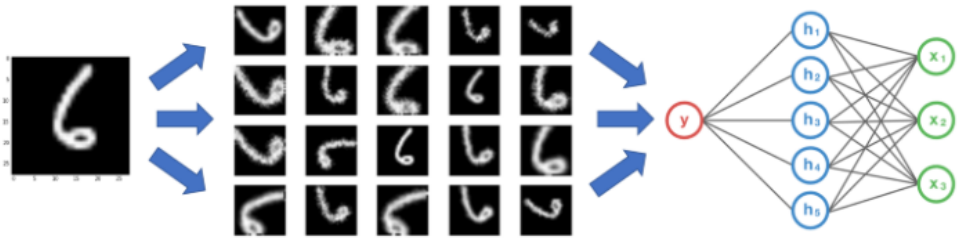

Figura 2.5: Ejemplo de las transformaciones que se aplican sobre una imagen de entrada al realizar *data augmentation*. (Fuente: [7])

A mayor número de épocas, más actualizaciones de los pesos se producen por lo que puede dar problemas de *overfitting* y en caso contrario de *underfitting* comentados en la sección 2.2.3

2.2.7. Data Augmentation

El aumento de datos o *data augmentation* es una técnica muy empleada para generar alteraciones artificiales sobre los datos originales y conseguir diversificar y aumentar los datos de entrenamiento de las redes neuronales (ver figura 2.5). El objetivo principal de esta técnica es tratar de combatir el *overfitting* y conseguir mejorar los resultados de la red al ser entrenada con más datos o transformaciones de los mismos o lo que es lo mismo, aumentar la robustez del sistema. El uso de *data augmentation* es común cuando los datos de entrenamiento son imágenes aplicando transformaciones típicas como rotaciones, giros horizontales o verticales y modificaciones en el brillo o contraste entre otros. No obstante, para otro tipo de entradas de datos también consigue mejorar el rendimiento del sistema como en el reconocimiento de voz al aplicar transformaciones a la onda de audio (velocidad, ruido ...)

2.3. Redes Neuronales Convolucionales (CNN)

Las redes neuronales convolucionales o *convolutional neural netwrok* (CNN) [8] son un tipo de red neuronal multicapa especialmente útiles cuando se dispone de datos con estructura espacial, por eso mismo es la red neuronal más usada a la hora de clasificar e identificar imágenes.

La arquitectura de este tipo de red neuronal está formada por dos etapas: extracción de características y clasificación (Ver figura 2.6). En la primera etapa la CNN es capaz de detectar en las imágenes de entrada desde patrones y características más simples como líneas o bordes hasta formas más complejas. El resultado de esta etapa es conocido como mapa de características o *feature maps* y son usados en la siguiente etapa donde una red totalmente conectada o fully connected (FC) seguida de una capa tipo softmax se encargan de combinar y clasificar las características.

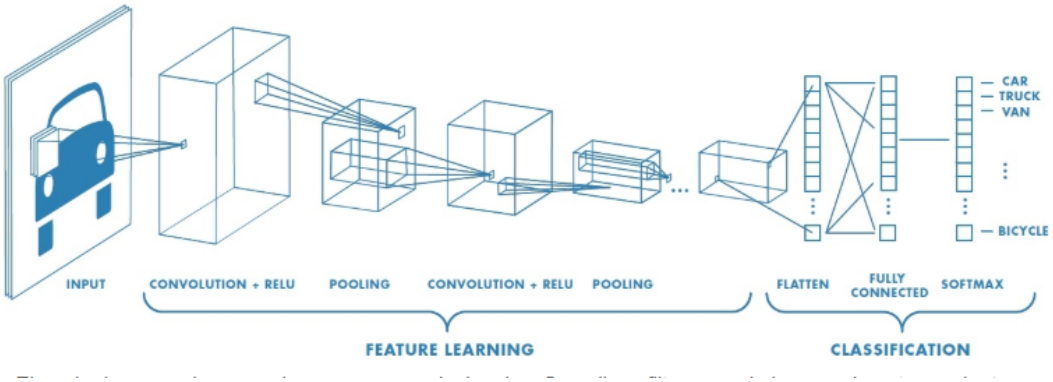

Figura 2.6: Esquema básica de una red neuronal convolucional. Extracción de características + clasificación (Fuente: [9])

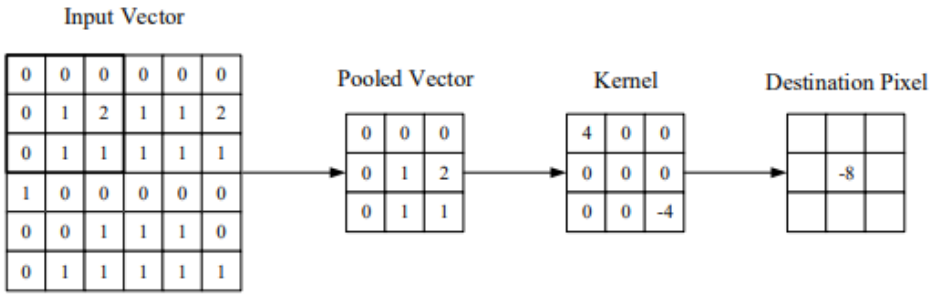

Figura 2.7: Proceso de la operación de convolución. Partes de la imagen inicial se convolucionan con un *kernel* 3x3 y el resultado se guarda en el centro de las partes escogidas de la imagen original (Fuente: [10])

2.3.1. Capas Convolucionales

Son las capas donde se realiza el mayor trabajo computacional debido al uso de convoluciones entre las imágenes de entrada y unos filtros llamados *kernels*. Los *kernels* tienen un tamaño mucho menor que la entrada y los pesos vienen determinado por su tamaño. Por ejemplo, usando un kernel de tamaño 3x3x3 el número de pesos de ese kernel será de 27. El valor de los pesos se ajusta de igual manera que en una red neuronal tradicional, durante el periodo de aprendizaje.

El proceso consiste en obtener la respuesta de los pixeles de entrada, que son la primera capa de neuronas de la red, en función de los *kernels* (producto escalar) recorriendo de forma iterativa toda la imagen de derecha a izquierda y de arriba abajo. Como resultado final se obtendrá por cada *kernel* un mapa de características donde cada pixel es una combinación lineal de los pixeles de entrada. El último paso de esta capa es aplicar la función de activación para incrementar la no-linealidad de la imagen y para este tipo de redes la función de activación más utilizada es la RELU (Ver sección 2.2.1) que básicamente cambia los valores negativos por el valor '0'.

El tamaño de salida se puede ajustar con 2 parámetros:

- Relleno o *Padding*: Añadir valores de '0' en los bordes de la imagen de entrada

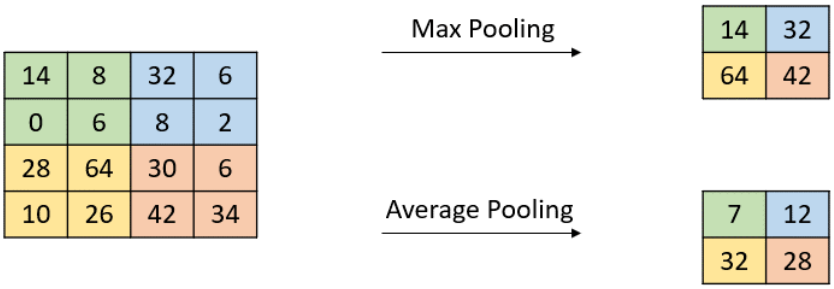

Figura 2.8: Ejemplo de las operaciones *max* y *average pooling* de tamaño 2x2

para conseguir que el kernel afecte a todos los pixeles de la entrada original. La imagen de salida de la capa de convolución sería del mismo tamaño al usar este tipo de *padding*.

- Paso o *Stride*: Se define como el salto que realiza el kernel en la imagen es decir, lo que avanza en cada iteración. Cuanto más alto es el stride menor información se analiza de la imagen original, pero se obtiene una imagen de salida más pequeña y disminuye el tiempo de procesamiento.

Al realizar la convolución típica se obtiene una imagen de menor tamaño que la original ya que no usa padding y su stride es igual a uno, pero de forma general se suele combinar estas dos opciones en función del objetivo y del tamaño de salida buscado de la capa de convolución.

2.3.2. Capas Pooling

Es una capa que va siempre después de cada capa de convolución ya que reduce las dimensiones de su salida antes de volver a pasar por otra capa de convolución. Su objetivo es mantener las características más importantes detectadas por cada filtro para que sirvan de entrada en las siguientes capas convolucionales y así reducir el coste computacional.

Las capas de pooling más típicas son (Ver figura 2.8):

- *Max poll*: Su funcionamiento es dividir la imagen de entrada en bloques o regiones de un tamaño específico y extraer por cada uno el máximo pixel.
- *Average poll*: Realiza la misma división por bloques pero esta vez se extrae la media calculada de los valores de los pixeles de cada bloque.

2.3.3. Capas Fully Connected

La salida final tras haber realizado todas las capas de convoluciones se conecta a unas capas totalmente conectadas (Ver figura 2.6). En estas capas todas las neuronas tienes conexiones entre ellas por capas y combinan todas las características aprendidas

por las capas anteriores para detectar patrones más grandes para su posterior clasificación.

La función de activación más corriente para problemas de clasificación es la función *softmax* que se aplica a la salida de las capas totalmente conectadas otorgando la probabilidad que tiene la entrada de ser cada tipo de clase o etiqueta (hay una neurona de salida por cada clase o etiqueta). Para ello, emplea la ecuación 2.6

$$p(y_j) = \frac{e^{y_j}}{\sum_{k=1}^{K}(e^{y_k}))} \tag{2.6}$$

en la que 'y' determina cada entrada a la última capa con *softmax* y 'K' determina el número de clases totales. Gracias a *softmax* se consigue resaltar las probabilidades mayores y penalizar las probabilidades pequeñas consiguiendo una mejor clasificación para las distintas clases.

Capítulo 3

Diseño y desarrollo

3.1. Datasets

3.1.1. Market-1501

Market-1501 [11] es un dataset enfocado para la re-identificación de personas ya que contiene 1501 identidades captadas por diferentes cámaras en frente de un supermercado de la Universidad Tsinghua en Pekín (Figura 3.1). En concreto, se usaron 5 cámaras de alta resolución (1280 x 1080 HD) y 1 cámara de baja resolución (720 x 576 SD) y al menos cada identidad fue capada por 2 o más cámaras diferentes. A partir de las imágenes captadas se consiguieron detectar los 32,668 bounding boxes usando la técnica DPM (*Deformable Part Model*) y no recortados a mano como en otros dataset usados para re-identificación de personas (ViPeR, CAVIAR...).

El dataset se encuentra dividido en dos conjuntos:

- Conjunto de entrenamiento (*Training set*): Cuenta con un total de 12936 imágenes correspondientes a 750 clases del total y son el conjunto de datos que sirven para entrenar y validar los modelos de redes neuronales.
- Conjunto de prueba (*Test set*): Cuenta con un total de 19732 imágenes correspondientes a 751 clases del total y son el conjunto usado para probar el funcionamiento de los distintos modelos una vez han sido entrenados. Además 3368 imágenes de este conjunto están destinados a usarse como consulta (*Query set*) para poder realizar labores de re-identificación

Los nombres de las imágenes determinan a que clase pertenecen, con qué cámara fueron tomadas y en que secuencia de dicha cámara se extrajeron. Por ejemplo el nombre de una de las imágenes es: 0005_c2s2_048412_01.jpg dónde '0005' determina la clase o etiqueta de la imágen, c2 determina la cámara y el resto del nombre indica la secuencia de video y el frame específico del video tomado con la cámara correspondiente.

Market-1501-Attributes

Es un dataset que contiene una serie de atributos anotados manualmente sobre Market-1501 [13]. Para cada identidad (1501) hay 27 atributos anotados que fueron especialmente seleccionados analizando el conjunto de datos para evitar un sesgado

Figura 3.1: Ejemplo del estilo de imágenes del dataset Market-1501. Se observan 2 imágenes de 12 identidades diferentes tomadas con diferentes cámaras (Fuente: [12])

muy elevado de los datos con cualquier atributo.

Los 27 atributos son: género (hombre, mujer), longitud del cabello (largo, corto), longitud de ropa inferior(largo, corto), tipo de ropa inferior (vestido, pantalón), largo de las mangas (largo, corto), llevar puesto sombrero (si, no), portar mochila (si, no), portar bolsa de mano (si, no), portar otro estilo de mochila (si, no), edad (niño, adolescente, adulto, anciano), 8 colores para la ropa superior (negro, blanco, rojo, morado, amarillo, gris, azul, verde) y 9 colores para la ropa inferior (negro, blanco, rosa, morado, amarillo, gris, azul, verde, marrón). Cada uno de los colores de la ropa es un atributo distinto y se evalúa de forma que si la imagen lo contiene o no. (Figura 3.2)

Se encuentran anotados en un fichero de datos tipo Matlab (.mat) de manera bi-

Attribute	Label
gender	2
hair	2
up	2
down	2
clothes	1
hat	1
backpack	1
bag	1
handbag	2
age	2
upwhite	2
downred	2

Figura 3.2: Ejemplo de como se encuentran asignados algunos de los atributos sobre una imagen de Market-1501. (Fuente: [13])

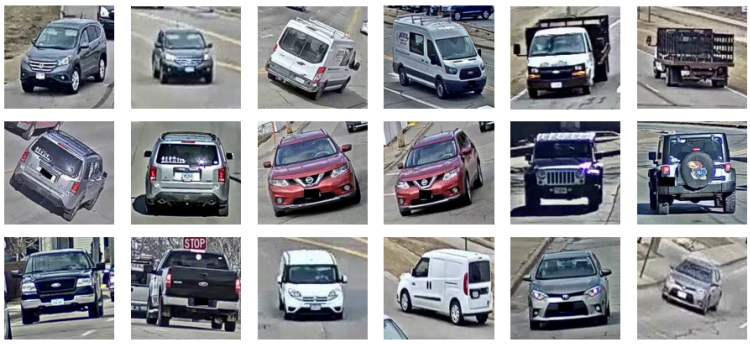

Figura 3.3: Ejemplo del estilo de imágenes del dataset Aicity. Se observan 2 imágenes de 9 identidades diferentes tomadas con diferentes cámaras

naria excepto el atributo edad, es decir, para cada atributo se encuentra el valor '1' si la identidad contiene dicho atributo o un tipo del mismo y el valor '0' en el caso contrario. En el caso del atributo edad encontramos cuatro opciones distintas: '1' niño, '2' adolescente, '3' adulto, '4' anciano.

3.1.2. Aicity

Aicity es un dataset que contiene imágenes de vehículos proporcionado por el *AI City Challenge* [14]. Este challenge lleva realizandose cada año desde 2017 con el objetivo de mejorar la inteligencia y seguridad de las ciudades con el análisis de videos de tráfico a través de distintos retos como la detección de anomalías de tráfico, la re-identificación de vehículos o el seguimiento (*tracking*) de los mismos.

Las imágenes de los vehículos fueron tomadas por 40 cámaras a lo largo de 10 intersecciones de una ciudad de Estados Unidos (ver figura 3.3). El dataset es uno de los más grandes destinados a los objetivos mencionados pero para este trabajo de fin de grado se ha usado solo una parte del dataset para tener un tamaño similar al dataset de personas.

Contamos con un total de 333 clases o identidades divididos entre los siguientes conjuntos:

- Conjunto de entrenamiento (*Training set*): Cuenta con un total de 18886 imágenes correspondientes a 166 clases del total y son el conjunto de datos que sirven para entrenar y validar los modelos de redes neuronales.
- Conjunto de prueba (*Test set*): Cuenta con un total de 17130 imágenes correspondientes a 167 clases del total y son el conjunto usado para probar el funcionamiento de los distintos modelos una vez han sido entrenados. Además 923 imágenes de este conjunto están destinados a usarse como consulta (*Query set*) para poder realizar labores de re-identificación.

Aicity-Attributes

Al igual que para el dataset de personas, se dispone de una serie de atributos para los vehículos de Aicity. Son 6 atributos y aunque la cantidad es bastante menor que el

Atributo	Etiqueta
Vehículo	Camioneta
Marca	GMC
Color	Blanco
Techo	No
Ventanas	Si
Orientación	6

Figura 3.4: Ejemplo de como se encuentran asignados los 6 atributos de una imagen de Aicity

anterior hay mas opciones en cada uno de los atributos.

Los 6 atributos son los siguientes (ver figura 3.4):

- Tipo de vehículo (8 opciones): Berlina, Todoterreno, Camión, Monovolumen, Camioneta, Autobús, Furgoneta y Otros,
- Marca (29 opciones): Dodge, Ford, Chevrolet, GMC, Honda, Chrysler, Jeep, Hyundai, Subaru, Toyota, Buick, KIA, Nissan, Volkswagen, Mustang, BMW, Cadillac, Volvo, Pontiac, Mercury, Lexus, Mercedes-Benz, Mazda, Scion, Mini, Lincoln, Audi, Mitsubishi y Otros.
- Color del vehículo (8 opciones): Negro, Blanco, Gris, Azul, Rojo, Oro, Verde y Amarillo
- Techo del Vehículo (3 opciones): Cubierto, Descapotable, Otros.
- Ventanas del vehículo (3 opciones): Sin ventanas, Con ventanas, Otros.
- Orientación del vehículo (8 opciones): Orientaciones marcadas con números de 0 a 7 con respecto a la cámara con que fueron tomadas las imágenes.

3.2. Pytorch

Pytorch es una librería desarrollada por Facebook en 2017 de código abierto basado en Python, especialmente útil en el ámbito del Deep Learning por disponer de una sencilla interfaz para el desarrollo de redes neuronales. Permite trabajar directamente con tensores y realizar los cálculos numéricos en la tarjeta gráfica o GPU. Esto es posible gracias al uso de una API desarrollada por NVIDIA llamada CUDA que conecta la CPU con la GPU. Este uso que Pytorch permite de la GPU posibilita la paralelización de los procesos, lo que conlleva una reducción en los tiempos de computación por eso es ideal a la hora de entrenar modelos

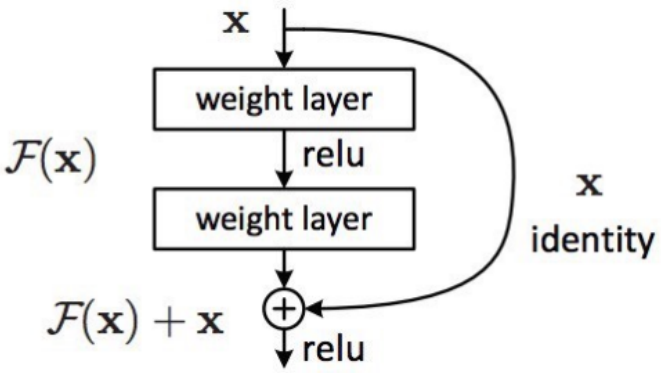

Figura 3.5: Bloque residual que añade una conexión de salto entre las capas de Resnet (Fuente: [15])

3.3. Transfer learning

La transferencia del aprendizaje o *transfer learning* es una de las técnicas más importantes de *deep learning* que consiste en el uso de redes neuronales que ya han sido entrenadas para una tarea y modificarlas para realizar otra tarea del estilo con otro conjunto de datos. Su principal ventaja es el ahorro de tiempo al no tener que entrenar una red desde cero sino que se toma como punto de partida una red pre-entrenada para aprovechar su aprendizaje de carácter general y adaptarla al nuevo objetivo.

Gracias a un gran dataset de imágenes llamado Imagenet existen numerosos modelos entrenados para clasificar dichas imágenes (clasificador de 1000 clases) y son usados como base para realizar tareas similares debido a que su tiempo de entrenamiento puede durar varias semanas según el hardware disponible. Para este tipo de tareas se pueden aprovechar la extracción de características de las imágenes (bordes, patrones...) para procesarlos posteriormente por un clasificador que si se entrenará desde cero. Por ello en este trabajo se ha optado por realizar la técnica de transferencia de aprendizaje sobre algunas de las redes neuronales convolucionales pre-entrenadas con Imagenet.

3.4. Redes Neuronales empleadas

3.4.1. Resnet

ResNet o *Residual Network* [15], es un tipo de red neuronal convolucional introducida por Microsoft en 2015 que como su nombre indica introduce un aprendizaje residual. ResNet incluye los llamados bloques residuales (Ver figura 3.5) los cuáles añaden una conexión de salto de capas que permite a la entrada agregarse a la salida de dichas capas para que la red aprenda la diferencia entre ellas. Gracias a estos bloques Resnet garantiza un punto de referencia a las capas del cuál aprender y soluciona el problema del desvanecimiento del gradiente que se produce al incrementar el número de capas de una red neuronal. Por ello, ResNet fue la primera red neuronal convolucional que aumentó el número de capas en sus modelos sin llegar a perder precisión por los problemas del gradiente mencionados. En este trabajo se ha usado una de sus arquitecturas más famosas: ResNet-50 donde el número indica la cantidad de capas de la red.

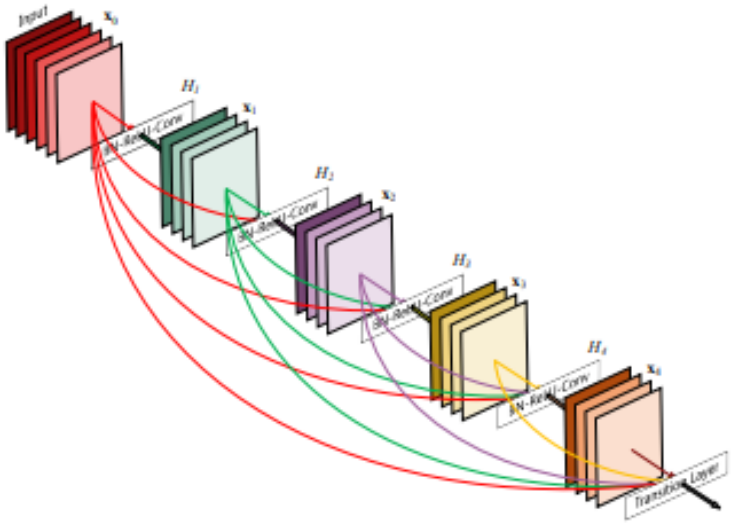

Figura 3.6: Dense Block de 5 capas. Cada capa recibe como entrada los mapas de características de las capas anteriores. (Fuente: [16])

3.4.2. Densenet

Densenet o *Dense Network* [16], es un tipo de red neuronal convolucional que consigue aumentar el número de capas evitando el problema del desvanecimiento del gradiente. Para ello, hace uso de unos bloques llamados *dense blocks* (Ver figura 3.6) formados por un determinado número de capas que se encuentran densamente conectadas entre ellas. Esto quiere decir que cada una de las capas recibe como entrada los mapas de características de las capas anteriores lo que mejora el flujo de información a lo largo de la red. El uso de estos bloques hace a la red Densenet una red muy compacta que no desperdicia las características aprendidas sino que las comparte para generar un conocimiento global.

3.5. Pre-Procesamiento

Los dataset Market-1501 y Aicity constan de 2 conjuntos de datos para entrenar y para comprobar el funcionamiento de los modelos, pero se ha realizado una división en la parte destinada al entrenamiento para evaluar el modelo. A este conjunto se le llama conjunto de validación y sirve para validar el modelo que se está entrenando es decir prevenir el *overfitting* y el *underfitting*.

Para ambos dataset se han usado 1 imagen de cada clase para evaluar el modelo, es decir 701 imágenes de validación para Market y 166 imágenes de validación para Aicity lo que corresponde al 5 % y 1 % de los datos de entrenamiento respectivamente.

Por su parte, los atributos de cada dataset se han dividido de igual manera para mantener la concordancia en el sistema y se han añadido a un fichero de texto en el que se describe la ubicación de cada imagen, su nombre y sus atributos correspondientes para su posterior procesado por un clasificador multiclase.

3.6. Desarrollo

Una vez preparado el conjunto de datos, haciendo uso de la técnica *transfer learning* se cargan las redes pre-entrenadas mencionadas anteriormente para su posterior entrenamiento con los conjuntos de datos (Market y Aicity). Además de cargar las redes, también se realiza un *fine tunning* de la capa de salida es decir de la capa totalmente conectada (*fully connected*) para seleccionar nuestro número de clases en vez de las 1000 establecidas en las redes pre-entrenadas.

Antes de entrenar los modelos es necesario seleccionar los valores de los hiperparámetros como la tasa de aprendizaje (*learning rate*), el número de épocas, el tamaño de lote (*batchsize*)... En el siguiente capítulo se realizarán variaciones en estos hiperparámetros para seleccionar los que mejor rendimiendo consigan.

Con el objetivo de realizar *data augmentation* se realizan transformaciones a las imágenes de los conjuntos de datos de entrenamiento y validación para crear modelos más robustos que generalicen el conocimiento. Algunas de estas transformaciones son el redimensionamiento del tamaño de las imágenes, el recorte aleatorio de ciertas zonas de la imágen, girar aleatoriamente la imágen horizontalmente y normalizar los valores de los pixeles entre otros.

Por último, se selecciona la función de pérdida o función de costes que empleará el modelo para poder realizar los algoritmos de aprendizaje, en este caso se empleará la función de coste *cross entropy loss* (Ver sección 2.2.4) por su rendimiento en clasificación de imágenes.

Una vez definidos los conceptos anteriores ya se procede a entrenar los modelos que se realizan en un bucle por épocas y en cada época se analiza las imágenes según el tamaño de batch definido. Por cada época se calcula la pérdida y la precisión para representarlo gráficamente y analizar el entrenamiento realizado. Una vez finalizado el entrenamiento, que tiene una duración estimada de unas 2-3 horas, se guarda el modelo para su posterior evaluación.

En la figura 3.7 se puede observar que el entrenamiento realizado sobre Densenet con los datos de Market-1501 ha sido positivo al seguir una evolución ideal de precisión y error. A medida que aumentan las épocas el error calculado va disminuyendo y la precisión va aumentando llegando a estabilizarse a partir de 20-30 épocas y aunque sea imposible observarse por la pequeña variación de valores, los resultados siguen mejorando progresivamente a medida que se aumentan las épocas.

Por otro lado, se ha entrenado un clasificador multiclase con la parte del dataset de entrenamiento. Para ello se realizan los mismos pasos que en el entrenamiento anterior (modelo, hiperparámetros ...) pero también se cargan los atributos de cada dataset ya que cada uno de los atributos corresponde con una clase del sistema (una clase por atributo).

El objetivo final es obtener por cada imágen de entrada su probabilidad de pertenecer a cada una de las clases, es decir la probabilidad que tiene de poseer cada uno

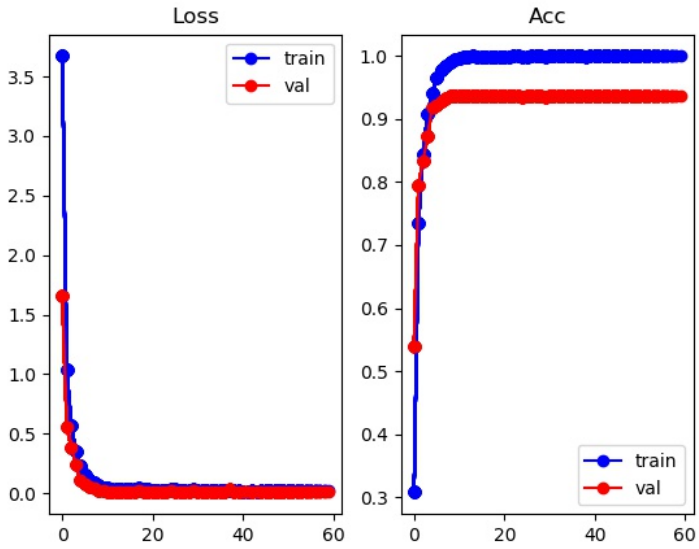

Figura 3.7: Resultado del entrenamiento de Resnet con Market-1501. Se ha usado una tasa de aprendizaje de 0.1 y un tamaño de lote de 64 imágenes por lote

de los atributos. La función de coste elegida sigue siendo la función *cross entropy loss* y el entrenamiento se realiza época a época teniendo en cuenta en cada una el tamaño del lote. Además, en este caso se cálcula la pérdida y la precisión para cada una de las clases (atributos) y no del sistema en global.

Capítulo 4

Evaluación

4.1. Introducción

En este capítulo se procederá a analizar las técnicas de evaluación de los modelos empleados así como distintas pruebas ajustando los hiperparámetros mas destacados. También se analizará que efecto tiene el empleo de los metadatos (atributos) al combinarlos con los algoritmos de re-identificación.

4.2. Marco de evaluación

La re-identificación de imágenes [17] [1] se puede ver como un problema de recuperación de imágenes ya que dada una imagen de consulta tomada con cierta cámara, el objetivo es encontrar otras imagenes del conjunto de test (conocido como galería) de la misma identidad que la consultada pero que hayan sido tomadas con cámaras distintas. En este caso las identidades son por una lado personas usando el dataset Market-1501 y por otro lado vehículos usando el dataset Aicity.

4.2.1. Algoritmos

Para poder realizar tareas de re-identificación es necesario calcular las distancias entre las imágenes colsultadas y las imágenes de la galería. Para ello todas las imágenes del conjunto de consulta (*query set*) y del conjunto de prueba o galería (*test set*) (ver sección 3.1) se hacen pasar por los modelos entrenados para extraer sus características. Dichas características, conocido como vector de *features* es extraído de la última capa de características de las redes neuronales convolucionales es decir, antes del clasificador (ver sección 2.3). Durante la extracción no solo se hace pasar la imagen correspondiente por el modelo sino que se pasa tanto la imagen como ella misma dada la vuelta. Con esta peculiaridad se obtiene por cada imagen dos vectores de características que se suman y se normalizan para obtener un solo vector algo más robusto por imagen. El resultado final se guarda en 2 matrices (para consulta y galería) de tamaño n x f donde n es el número de imágenes de cada conjunto y f el número de características de cada uno.

Una vez extraídos los vectores de características, se calculan las distancias de re-identificación. Básicamente se trata de calcular la distancia entre cada vector de características de las consultas respecto toda la galería y para ello se utiliza la similitud coseno. Al tener distancias normalizadas, para calcular la similitud coseno es suficiente

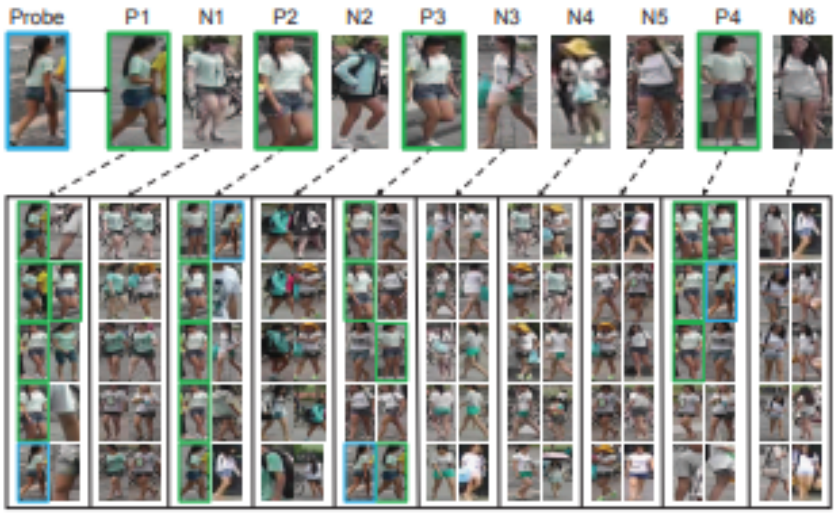

Figura 4.1: Ejemplo de los vecinos más cercanos en un sistema de re-identificación. En la parte superior se observa la imagen consultada y sus 10 vecinos más cercanos. La parte inferior refleja los 10 vecinos más cercanos de cada una y las posibles coincidencias con la consulta (Fuente: [18])

con calcular la multiplicación matricial entre las matrices de características de consulta y galería y su posterior transposición. Como resultado se obtiene una matriz que representa la distancia de predición que hay para cada imagen de consulta sobre la galería que se ordena de menor a mayor similitud para su posterior evaluación.

Una técnica opcional conocida como *re-ranking* [18] para calcular de otra forma la similitud a partir de la extración de características. La idea que persigue esta técnica es que si una imagen de la galería es similar a la imagen de consulta en los vecinos recíprocos más cercanos, es más probable que sea una estimación positiva o correcta de esa consulta.

Entrando en detalle, se calculan los k-vecinos más próximos para cada imagen de consulta (sobre la galería) y a partir de estos se calcula la distancia de Jaccard para poder calcular de nuevo la distancia de cada consulta sobre la galería es decir, reclasificarlos (*re-ranking*). A más vecinos recíprocamente cercanos tengan una imágen de consulta con otra de galería, más similitud tendrán por la distancia de Jaccard. El funcionamiento de la distancia de Jaccard es asignar mayor o menor similitud entre las imágenes de consulta y galería según la cantidad de vecinos recíprocamente cercanos tengan entre ellos, a más duplicados o coincidencias mayor es su similitud y viceversa.

Por último, para calcular la distancia final se tiene en cuenta tanto la distancia original como la distancia de Jaccard siguiendo la fórmula 4.1:

$$d_F(q, g_i) = (1 - \lambda) * d_J(q, g_i) + \lambda * d(q, g_i) \tag{4.1}$$

donde d y d_J son la distancia original y la distancia de Jaccard respectivamente y λ un parámetro que asigna una importancia o peso en la operación a cada distancia. En nuestro caso se han usado un valor de k=20 es decir tener en cuenta los 20 vecinos más cercanos y un valor de $\lambda = 0.3$ lo que implica que damos más importancia a la distancia

de Jaccard calculada pero sin llegarnos a olvidar del todo de la distancia original.

4.2.2. Métricas

Para evaluar los modelos entrenados se han empleado dos de las métricas más usadas en tareas de re-identificación de imágenes como son el *Cumulative Matching Characteristics* (CMC) y el *mean Average Precision* (mAP).

La medida CMC [19] sirve para analizar el rendimiento del algoritmo de re-identificación sobre el conjunto de datos de test y de consulta. Realiza una clasificación de las muestras de test de acuerdo a la distancia (similitud) respecto la imágen consultada ordenandolos de menor a mayor distancia. Para cada una de las imágenes que se usan como *query set* se calcula el CMC siguiendo la siguiente fórmula:

$$CMC_k = \begin{cases} 1 & \text{Si el top-K de muestras de test contienen la identidad de consulta} \\ 0 & \text{Si la identidad consultada no se encuentra en el top-K de muestras} \end{cases} \tag{4.2}$$

Por lo tanto el CMC_k mide la probabilidad de encontrar una coincidencia correcta en el top_k de mejor similitud. Se considera que cuanto mejor es el algoritmo o sistema evaluado más alto será su CMC y a medida que aumentan los datos de test menor será su valor debido al aumento de posibilidades erróneas.

Para evaluar los modelos al completo se ha realizado el promedio del CMC sobre todas las imágenes de consulta obteniendo así un valor promedio de acierto que comprende valores entre 0 y 1.

Por su parte, la medida mAP se trata de una media de la precisión promedio (AP) [20]. Se trata de una medida ideal para sistemas que devuelven una secuencia ordenada de datos, en nuestro caso las mejores coincidencias para cada imágen consultada, debido a que el AP tiene en cuenta tanto la precisión del sistema como el orden de aparación de los casos correctos. Para calcular la precisión promedio (AP) de cada imagen de consulta con respecto al conjunto de prueba se hace uso de la siguiente fórmula:

$$ap = d_{recall} * Acc \tag{4.3}$$

donde d_{recall} indica la sensibilidad (*recall*) y Acc la precisión del sistema. La sensibilidad aumenta a medida que se van encontrando casos correctos por lo que según la aparación de dichos casos la precisión promedia aumentará a mayor o menor velocidad que tomando únicamente la presición (Acc).

Para la evaluación de los modelos se ha tomado el mAP es decir la media global sobre cada precisión promedio de las imágenes de consulta.

4.3. Pruebas y resultados

A continuación se listan los resultados de los experimentos realizados. Como primer paso se ha realizado un estudio para ajustar algunos de los hiperparámetros más importantes de los modelos descritos en la sección 3.4 teniendo en cuenta tanto su *accuracy* y *loss* en la fase de entrenamiento, como los resultados obtenidos en labores

MARKET-1501 / RESNET					
		Top1		mAP	
Learning Rate	**BatchSize**	**Normal**	**Re-Ranking**	**Normal**	**Re-Ranking**
0.001	**16**	0.786817	0.831354	0.554239	0.745435
	32	0.744359	0.794834	0.512038	0.693142
	64	0.699228	0,742577	0.453468	0.623282
0.01	**16**	**0.875594**	**0.896378**	0.690974	0.837935
	32	0.839074	0.871734	0.640974	0.801981
	64	0.800178	0.846496	0.583698	0.756510
0.1	**16**	0.825416	0.860451	0.614715	0.790704
	32	0.872625	0.895784	**0.706056**	**0.839231**
	64	**0.878563**	**0.898159**	**0.709395**	**0.839944**

Tabla 4.1: Resultados de re-identificación usando Resnet sobre el conjunto de datos de Market-1501. Los dos mejores resultados para cada métrica se encuentran marcados en verde

de re-identificación y dicho estudio se ha realizado para ambos dataset con el objetivo de seleccionar los que mejor rendimiento tengan para su posterior combinación con los metadatos.

Al no existir unos valores específicos de los hiperparámetros que se consideren correctos para las redes neuronales, se ha optado por variar tanto la tasa de aprendidaje (*learning rate*) como el tamaño del batch (*BatchSize*) empleado. Previamente se ha probado con una tasa de aprendizaje de 0.01 y un batchsize de 32 y se ha observado que tenian un rendimiento aceptable por lo que se han seleccionado valores típicos y a la vez cercanos a estos para realizar las variaciones. Además se ha optado por mantener un número de épocas fijo (60 épocas) para asegurar la convergencia de los datos en todos las variaciones mencionadas ya que al disponer de GPU el entrenamiento de los modelos suponen no más de 2-3 horas cada uno.

4.3.1. Modelos con Market-1501

A la vista de los resultados otorgados por la tabla 4.1 se puede concluir que tanto aplicando *re-ranking* o sin aplicarlo, los mejores resultados para la red Resnet se obtienen entrenando el modelo con mayor tasa de aprendizaje (0.1) y usando un tamaño de batch de 64 por lo que será el seleccionado para combinarlo posteriormente con los metadatos. También destaca que usando una tasa de aprendizaje menor (0.01) y un tamaño de batch de 16 se obtienen buenos resultados en el Top1 (métrica CMC para la primera predicción) sin embargo su *mean Average Precision* (mAP) no llega a ser uno de los mejores resultados. En general, también se puede observar que al aplicar la técnica *re-ranking* se obtienen mejores resultados que si se opta por no usarla.

La tabla 4.2 muestra los resultados del mismo experimento realizado anteriormente manteniendo el conjunto de datos de Market-1501 pero usando la red neuronal convolucional Densenet. Los resultados evidencian un comportamiento parecido al anterior

MARKET-1501 / DENSENET					
		Top1		mAP	
Learning Rate	**BatchSize**	**Normal**	**Re-Ranking**	**Normal**	**Re-Ranking**
0.001	**16**	0.882435	0.839667	0.647616	0.752030
	32	0.858670	0.794537	0.661649	0.692124
	64	0.888655	0.745249	**0.729963**	0.619254
0.01	**16**	0.884204	**0.902316**	0.707228	**0.853028**
	32	0.850059	0.885986	0.647040	0.819930
	64	0.811758	0.854513	0.586314	0.770525
0.1	**16**	0.882435	0.879751	0.647616	0.820840
	32	**0.888670**	0.883017	0.661649	0.822922
	64	**0.888658**	**0.910036**	**0.729963**	**0.865975**

Tabla 4.2: Resultados de re-identificación usando Densenet sobre el conjunto de datos de Market-1501. Los dos mejores resultados para cada métrica se encuentran marcados en verde

ya que los mejores resultados en CMC y mAP se obtienen aplicando la mayor tasa de aprendizaje usada (0.1) y el mayor tamaño de batch empleado (64). Los resultados experimentan una mejoría cuando se opta por utilizar la técnica de *re-ranking* por lo que siguen en la línea de los resultados anteriores. Cabe mencionar que empleando una tasa de aprendizaje de 0.01 y un tamaño de batch de 16 se obtienen buenos resultados en Top1 y al igual que en el caso anterior en mAP no consiguen a llegar a ser de los mejores resultados.

4.3.2. Modelos con Aicity

Se han realizado los mismos experimentos con ambos modelos pero esta vez usando el dataset de coches para seleccionar aquellos hiperparámetros que mejor rendimiento de re-identificación proporcionen.

Usando la red neuronal convolucional Resnet se han obtenido los resultados expuestos por la tabla 4.3. Esta vez el mejor resultado viene dado al entrenar el modelo con la tasa de aprendizaje intermedia (0.01) y el menor de los tamaños de batch (16). Se consiguen unos resultados similares, aunque algo inferiores, empleando la mayor tasa de aprendizaje y el mayor tamaño de batch que casualmente, son los valores que mejor funcionaban con el dataset MArket-1501. Por otro lado, se puede observar que los resultados obtenidos al usar la técnica *re-ranking* son bastntes superiores en este caso sobre todo en la métrica mAP.

Cambiando la red neuronal convolucional a la red Densenet se obtienen unos resultados similares que se ven reflejados en la tabla 4.4. La mejor combinación de hiperparámetros sigue siendo al emplear el menor tamaño de batch (16) y la tasa intermedia de aprendizaje (0.01). Al igual que en el caso anterior, usando la mayor tasa de aprendizaje y tamaño de batch se alcanzan resultados parecidos aunque mínimamente inferiores. Siguiendo la tendencia de los demás experimentos, empleando el algortimo de *re-ranking* se obtienen mejores resultados que si se escoge no usarlo.

AICITY / RESNET					
		Top1		mAP	
Learning Rate	BatchSize	Normal	Re-Ranking	Normal	Re-Ranking
0.001	16	0.595005	0.606949	0.296210	0.330859
	32	0.565689	0.598263	0.268092	0.302705
	64	0.523344	0.565689	0.244432	0.278938
0.01	16	**0.692725**	**0.706840**	**0.386364**	**0.429138**
	32	0.663409	**0.675353**	0.346318	0.385604
	64	0.624321	0.634093	0.322956	0.358928
0.1	16	0.333333	0.327904	0.151647	0.169715
	32	0.509224	0.504886	0.258014	0.294562
	64	**0.678610**	0.660152	**0.378085**	**0.423384**

Tabla 4.3: Resultados de re-identificación usando Resnet sobre el conjunto de datos de Aicity. Los dos mejores resultados para cada métrica se encuentran marcados en verde

4.3.3. Conclusiones de las primeros análisis

Una vez evaluados los modelos y analizados los resultados de las tablas 4.1, 4.2, 4.3 y 4.4 se pueden sacar las siguientes conclusiones:

- Tanto empleando tasas de aprendizaje de 0.01 y 0.1 y tamaños de bacth de 16 y 64 respectivamente se obtienen los mejores resultados para ambos conjuntos de datos y redes neuronales.
- Empleando el algoritmo de *re-ranking* se mejora el rendimiento final de los distintos modelos.
- Utilizando la red neuronal convolucional Densenet se consiguen unos resultados por norma general mejores que usando Resnet en estos casos.
- Los resultados empleando el conjunto de datos de personas son mejores que usando los de vehículos, probablemente por la mayor variabilidad en los vehículos y la mayor cantidad de cámaras empleadas.

4.3.4. Análisis combinado

Como análisis objetivo y final se ha realizado una combinación de las distancias de re-identificación base con las distancias de re-identificación generadas a partir de los atributos de los conjuntos de datos de Market-1501 y Aicity. Para ello se han seleccionado los modelos entrenados que mejor rendimiento tienen (uno para cada red neuronal convolucional sobre cada dataset) para usar sus vectores de características como punto da partida.

Se ha entrenado el clasificador multiclase con los atributos de ambos conjuntos de datos (personas y vehículos) por separado para extraer tanto el vector de características como el vector de probabilidad de cada atributo. Para obtener el vector de características, una vez finalizado el entrenamiento, se emplean los datos y atributos del conjunto de test y consulta para pasarlos por el modelo y extraer de la última capa antes del

AICITY / DENSENET					
		Top1		mAP	
Learning Rate	**BatchSize**	**Normal**	**Re-Ranking**	**Normal**	**Re-Ranking**
0.001	**16**	0.606949	0.636265	0.302012	0.337654
	32	0.563518	0.589577	0.265685	0.298903
	64	0.532030	0.578719	0.242233	0.278736
0.01	**16**	**0.725299**	**0.739414**	**0.398328**	**0.440754**
	32	0.694897	**0.713355**	0.369115	0.410694
	64	0.642780	0.661238	0.327798	0.367502
0.1	**16**	0.371336	0.370250	0.167484	0.184924
	32	0.542588	0.545060	0.261186	0.297293
	64	**0.688382**	0.706840	**0.382012**	**0.428934**

Tabla 4.4: Resultados de re-identificación usando Densenet sobre el conjunto de datos de Aicity. Los dos mejores resultados para cada métrica se encuentran marcados en verde

clasificador el vector de características. Para el vector de probabilidad simplemente se coge la salida del clasificador, que proporciona la probabilidad que tiene cada dato de entrada de tener cada uno de los atributos.

Una vez obtenidos los vectores de características y probabilidad empleando los metadatos (atributos de los datos) se han generado distancias de re-identificación y como era de esperar, usando solo esos vectores, los resultados son muy inferiores al estar entrenados por atributos y no por las distintas identidades. A pesar de eso, el objetivo es comprobar si esas características extras son capaces de mejorar los resultados y por eso se ha procedido a combinarlos con los vectores del sistema de re-identificación "original". Varias de las pruebas realizadas no mejoraban los resultados e incluso los empeoraban, algunas de ellas fueron:

- Concatenar directamente los vectores de características (inicial + metadatos) para la consulta y galería es decir obtener un vector de características de mayor tamaño para ambos conjuntos y posteriormente calculas las distancias entre las consultas y galería empleando dichos vectores de mayor tamaño.
- Concatenar de igual manera los vectores de características del sistema inicial con las probabilidades extraídas por el clasificador multi clase para el conjunto de consulta y la galería de test y usarlos para generar la distancia entre ambos conjuntos.

La idea que más se aproximó al objetivo buscado y que incluso consiguió realizarlo con una leve mejoría fue la de combinar directamente los distancias de características del conjunto de consulta (query set) y del conjunto de prueba (test set) para su posterior ordenamiento y evaluación. Se calculan 2 distancias

- Distancia re-id inicial: La distancia de cada consulta con la galería usando los vectores de características del sistema de re-identificación inicial.
- Distancia re-id metadatos: La distancia de cada consulta con la galería usando los vectores de características del clasificador multiclase (atributos).

MARKET-1501 / RESNET (0.1, 64)					
Pesos		**Top1**		**mAP**	
RE-ID	**Metadatos**	**Normal**	**Re-Ranking**	**Normal**	**Re-Ranking**
100 %	**0 %**	**0.878563**	**0.898159**	**0.709395**	**0.839944**
50 %	**50 %**	0.852827	0.889067	0.650357	0.812799
75 %	**25 %**	0.875423	0.897473	0.704840	0.836081
90 %	**10 %**	0.884470	**0.905879**	**0.718535**	0.849621
95 %	**5 %**	**0.880204**	0.905583	0.717019	**0.849992**
99 %	**1 %**	0.876720	0.904988	0.713246	0.849941

Tabla 4.5: Resultados de re-identificación al combinar con distintos pesos las distancias de re-identificación del sistema original (Resnet con Market-1501) y los metadatos. Los valores originales se encuentran marcados en azul y la mejor combinación para cada métrica en verde

Para combinar las distancias se experimentó con distintos pesos que otorgaban una importancia determinada a cada una de las distancias para ver la combinación que mejor rendimiento proporcionase.

La ejecución de este análisis para las imágenes y atributos del dataset Market-1501 se puede observar en las tablas 4.5 y 4.6. Para ambos modelos se ha usado la tasa de aprendizaje de 0.1 y el tamaño de batch de 64 (imágenes por lote) cuyo resultado original (100 % Re-Id) se muentra en color azul.

Examinando la evolución de los datos empleando las características extraídas por Resnet, se puede concluir que asignando un peso a las características de los metadatos entre el 1 % y 10 % aproximadamente, se consigue mejorar levemente el rendimiento tanto en CMC como en mAP. Sin embargo si se le otorga mayor importancia de ese porcentaje los resultados comienzan a empeorar y por lo tanto no cumpliría el objetivo. El mayor rendimiento en CMC y en Top1 se alcanzaría al emplear un 10 % de importancia a los metadatos mientras que para la métrica de avaluación mAP bastaría con un 5 % de importancia.

Por otro lado, en el caso de haber empleado Densenet, el análisis sigue la misma tendencia, mejorando si la importancia de los metadatos se encuentra aproximadamente entre el 1 % y 10 % y empeorando a medida que aumentamos su peso. En esta situación, el mayor rendimiento se alcanza al emplear el 5 % de importancia para los metadatos para ambas métricas de evaluación.

Para ambos casos las mejoras producidas son en torno al 1 % y empleando la técnica *re-ranking* se siguen obteniendo mejores resultados que sin emplearse aunque en el caso de la combinación la mejora sigue la misma línea se use o no esta técnica.

En las tablas 4.7 y 4.8 se puede estudiar el efecto de la combinación realizada sobre el dataset Aicity empleando los modelos con tasa de aprendizaje 0.01 y tamaño del batch de 16 (imágenes por lotes). Se puede observar que al igual que el caso anterior, los resultados iniciales (marcados en azul) se ven levemente mejorados al aplicarles una importancia a las características de los metadatos de entre el 1 % y 10 % aproxi-

MARKET-1501 / DENSENET (0.1, 64)					
Pesos		Top1		mAP	
RE-ID	Metadatos	Normal	Re-Ranking	Normal	Re-Ranking
100 %	0 %	0.888658	0.910036	0.729963	0.865975
50 %	50 %	0.862031	0.908083	0.673796	0.839939
75 %	25 %	0.884893	0.911943	0.720105	0.862715
90 %	10 %	0.892299	0.917755	0.732662	0.874694
95 %	5 %	0.893705	0.918349	0.737674	0.874944
99 %	1 %	0.892221	0.917755	0.737122	0.874783

Tabla 4.6: Resultados de re-identificación al combinar con distintos pesos las distancias de re-identificación del sistema original (Densenet con Market-1501) y los metadatos. Los valores originales se encuentran marcados en azul y la mejor combinación para cada métrica en verde

AICITY / RESNET (0.01, 16)					
Pesos		Top1		mAP	
RE-ID	Metadatos	Normal	Re-Ranking	Normal	Re-Ranking
100 %	0 %	0-692725	0.706840	0.386364	0.429138
50 %	50 %	0.534202	0.408252	0.260896	0.192120
75 %	25 %	0.674267	0.682953	0.374616	0.379257
90 %	10 %	0.689468	0.716612	0.389373	0.429596
95 %	5 %	0.699240	0.711183	0.389004	0.431893
99 %	1 %	0.692725	0.706840	0.386891	0.429984

Tabla 4.7: Resultados de re-identificación al combinar con distintos pesos las distancias de re-identificación del sistema original (Resnet con Aicity) y los metadatos. Los valores originales se encuentran marcados en azul y la mejor combinación para cada métrica en verde

madamente tanto en CMC como en mAP. Al aplicar una importancia superior a las distancias generadas por los metadatos se obtienen peores resultados según se aumenta de importancia y en este caso los resultados empeoran a mayor velocidad que en el caso anterior usando el dataset Market-1501 lo que se puede observar al aplicar un 50 % de importancia. Los mejores resultados empleando *re-ranking* para este caso (marcados de color verde) y para ambos modelos (Resnet y Densenet) se consiguen aplicando un 10 % para CMC y un 5 % para mAP. Sin usar *re-ranking*, ocurre lo contrario para los dos modelos es decir aplicando un 10 % de importancia se consigue el mejor resultado para mAP y con un 5 % para CMC.

4.3.5. Pruebas visuales

En esta sección se van a evaluar visualmente los resultados obtenidos al realizar la combinación con el objetivo de observar y demostrar la leve mejoría.

Se van a generar las 8 primeras predicciones para cada consulta antes y después de realizar la combinación para observar el cambio. Concretamente, se realiza el proceso sobre los mejores resultados obtenidos en la sección 4.3.1 y 4.3.2 con los resultados de

AICITY / DENSENET (0.01, 16)					
Pesos		**Top1**		**mAP**	
RE-ID	**Metadatos**	**Normal**	**Re-Ranking**	**Normal**	**Re-Ranking**
100 %	**0 %**	**0.725299**	**0.739414**	**0.398328**	**0.440754**
50 %	**50 %**	0.552660	0.426710	0.253286	0.184239
75 %	**25 %**	0.717698	0.710098	0.385112	0.380948
90 %	**10 %**	0.720955	**0.743757**	**0.401801**	0.440958
95 %	**5 %**	**0.729642**	0.740499	0.400934	**0.443992**
99 %	**1 %**	0.727470	0.739414	0.398948	0.441793

Tabla 4.8: Resultados de re-identificación al combinar con distintos pesos las distancias de re-identificación del sistema original (Densenet con Aicity) y los metadatos. Los valores originales se encuentran marcados en azul y la mejor combinación para cada métrica en verde

la sección 4.3.2. Por lo tanto, se selecciona los mejores parámetros para Densenet como modelo para ambos conjuntos de datos y también se seleccionan las distancias de re-identificación proporcionadas por la técnica de *re-ranking* ya que otorgan los mejores resultados. Aún así se ha comprobado que con los mejores parámetros del modelo Resnet y sin usar la técnica de *re-ranking* se producen resultados prácticamente idénticos (en el top-8) por lo que el análisis sobre las pruebas visuales que se proporcionan a continuación son válidos para sacar conclusiones generales.

Se han seleccionado aquellas consultas que en su top-8 predicciones se puede observar algún tipo de cambio ya que en muchas de ellas no hay variación o no es suficiente observando solamente el top-8 y se necesitaría realizar un análisis más profundo.

Además, que se consigan mejorar los resultados generales sobre todas las consultas no significa que en cada una de ellas se haya producido una mejora por lo que es probable que en algún caso aislado, al realizar la combinación, se empeore mínimamente sus resultados. Aún así, el caso más probable es observar una mejora sobre todo en las primeras predicciones y es lo que se va a probar visualmente.

Para el caso de las personas (Market-1501), se han extraído las figuras 4.2 y 4.3 que representan los resultados de re-identificación antes y después de la combinación respectivamente. Se puede observar que originalmente el sistema detectaba 4 aciertos a lo largo de las 8 predicciones situadas en posiciones intermedias y una vez aplicado la combinación se han conseguido acertar 5 de las 8 predicciones. Además la posición del primer acierto corresponde con la primera predicción cosa que no pasaba originalmente por lo tanto se puede entender la pequeña mejora tanto en el Top-1 (CMC) como en mAP.

Con un solo ejemplo no se puede demostrar al 100 % los resultados obtenidos en las tablas de las secciones anteriores, ya que en ellas se extraen resultados generales de todas las consultas, pero sirve para aportar un ejemplo visual de la existencia de cierta mejoría, que es lo que reflejan los datos de las tablas.

Para el caso de los coches (Aicity), se han extraído las figuras 4.4 y 4.5 que repre-

Figura 4.2: Ejemplo visual de los resultados de re-identificación originales para personas. Se observan las 8 primeras predicciones de izquierda a derecha para la consulta dada (*query*). Las predicciones correctas se encuentran marcadas de color verde y las erróneas de color rojo

Figura 4.3: Ejemplo visual de los resultados de re-identificación combinados para personas. Se observan las 8 primeras predicciones de izquierda a derecha para la consulta dada (*query*). Las predicciones correctas se encuentran marcadas de color verde y las erróneas de color rojo

Figura 4.4: Ejemplo visual de los resultados de re-identificación originales para vehículos. Se observan las 8 primeras predicciones de izquierda a derecha para la consulta dada (*query*). Las predicciones correctas se encuentran marcadas de color verde y las erróneas de color rojo

Figura 4.5: Ejemplo visual de los resultados de re-identificación combinados para vehículos. Se observan las 8 primeras predicciones de izquierda a derecha para la consulta dada (*query*). Las predicciones correctas se encuentran marcadas de color verde y las erróneas de color rojo

sentan los resultados de re-identificación antes y después de la combinación respectivamente. Se puede observar que originalmente el sistema detectaba 4 aciertos a lo largo de las 8 predicciones situadas en las posiciones 2,4,5 y 8. Al aplicar la combinación se consiguen 6 resultados en este caso y también se consiguen predicciones positivas más tempranas al estar los 6 aciertos en las posiciones 2,3,4,5,6 y 7. Este sería un caso que para los resultados generales no aportaría una mejora en el top-1 (CMC) pero si sería importante para la mejora producida en mAP.

Al igual que ocurre con las personas, con un solo ejemplo no se puede demostrar al 100 % los resultados obtenidos en las tablas de las secciones anteriores, pero si implican una prueba de la existencia de algún tipo de mejora.

Capítulo 5

Conclusiones y trabajo futuro

5.1. Conclusiones

Durante este trabajo de fin de grado se han realizado varios estudios que han contribuido a realizar el objetivo principal del trabajo.

La realización del estado del arte ha resultado de bastante utilidad ya que ha facilitado la comprensión del trabajo al estudiar el funcionamiento y los conceptos básicos del aprendizaje profundo en redes neuronales en especial en las redes neuronales convolucionales para el tratamiento de imagenes como datos de entrada.

El objetivo principal del proyecto es el de analizar el uso de distintos modelos de redes neuronales convolucionales en tareas de re-identificación de imágenes. Se ha optado por utilizar redes pre-entrenadas como Resnet y Densenet y se han utilizado dos dataset para realizar las pruebas sobre datos de distintos tipos, uno contiene imágenes de personas llamado Market-1501 y el otro vehículos de todos los tipos llamado Aicity. Gracias a que estos dataset disponían de atributos que complementaban la información de sus datos se ha podido ver el efecto que tiene el uso de estos metadatos en tareas de re-identificación de imagenes.

Los resultados obtenidos para cada dataset han sido positivos llegando aproximadamente hasta un 85 % de precisión media en el conjunto de datos de personas y de casi un 50 % en el caso de los vehículos. Además el Top1 de prediciones para el caso de personas llega al 90 % (es decir 9 de cada 10 primeras prediciones son correctas) mientras que para vehículos el Top1 llega al 70 %.

A partir de un clasificador multiclase se han entrenado los atributos de cada dataset para obtener a partir de ellos características extra de los dataset y así poder combinarlos con los resultados originales. El resultado ha sido una mejoría en torno al 1 % y 2 % para personas y de casi un 1 % para vehículos tanto en primeras predicciones como en precisión mediaa global. Aunque la mejora es bastante pequeña, el objetivo se ha cumplido ya que sirve para comprobar que el empleo de metadatos puede mejorar los resultados de re-identificación y sirve de punto de partida para futuras mejoras.

5.2. Trabajo futuro

A la vista de los resultados que se han obtenido en este trabajo se propone las siguientes medidas para intentar mejorar los resultados u obtener nuevas vías de análisi:

- Emplear otro tipo de redes pre-entrenadas para observar si los resultados siguen la línea de los usados en este trabajo o incluso emplear redes con mas capas es decir redes más profundas.
- Entrenar los mismos modelos pero cambiando o añadiendo transformaciones de *Data Augmentation* o variando otros hiperparámetros o los mismos empleados en el trabajo de distinta manera. También se podría analizar el efecto de realizar cambios en los conjuntos de entrenamiento, validación y sobre todo de los conjuntos de consulta y galería para observar el comportamiento de los resultados al realizar otro tipo de consultas.
- Entrenar otro tipo de clasificador para los atributos o entrenar con el mismo modelo pero variando realizando un estudio variando sus hiperparámetros más importantes.
- Estudiar posibles combinaciones entre las características extraídas por el clasificador multiclase y el sistema original que puedan dar a una mayor mejora en los resultados que los conseguidos en este trabajo

Bibliografía

[1] S. Karanam, M. Gou, Z. Wu, A. Rates-Borras, O. I. Camps, and R. J. Radke, "A systematic evaluation and benchmark for person re-identification: Features, metrics, and datasets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, pp. 523–536, 2019. 1, 25

[2] F. Vázquez, "Deep learning fácil con deepcognition." `https://planetachatbot.com/deep-learning-facil-con-deepcognition-9af43b2319ba`, Enero 2018. Accedido en febrero de 2020. 8

[3] A. Ghasemian, H. Hosseinmardi, and A. Clauset, "Evaluating overfit and underfit in models of network community structure," *ArXiv*, vol. abs/1802.10582, 2018. 9

[4] A. Bhande, "What is underfitting and overfitting in machine learning and how to deal with it.." `https://medium.com/greyatom/what-is-underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-`Marzo 2018. Accedido en febrero de 2020. 9

[5] P. Golik, P. Doetsch, and H. Ney, "Cross-entropy vs. squared error training: a theoretical and experimental comparison," pp. 1756–1760, 08 2013. 10

[6] B. D. Hammel, "What learning rate should i use?." `http://www.bdhammel.com/learning-rates/`, Marzo 2019. Accedido en marzo de 2020. 11

[7] A. Gandhi, "Data augmentation. how to use deep learning when you have limited data." `https://nanonets.com/blog/data-augmentation-how-to-use-deep-learning-when-you-have-limited-data-part-2/`, 2018. Accedido en marzo de 2020. 12

[8] C.-C. J. Kuo, "Understanding convolutional neural networks with a mathematical model," *J. Vis. Commun. Image Represent.*, vol. 41, pp. 406–413, 2016. 12

[9] MathWorks, "Redes neuronales convolucionales." `https://es.mathworks.com/solutions/deep-learning/convolutional-neural-network.html`, 2017. Accedido en marzo de 2020. 13

[10] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," *ArXiv e-prints*, 11 2015. 13

[11] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian, "Scalable person re-identification: A benchmark," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015. 17

[12] S. Li, X. Liu, W. Liu, H. Ma, and H. Zhang, "A discriminative null space based deep learning approach for person re-identification," 08 2016. 18

[13] Y. Lin, L. Zheng, Z. Zheng, Y. Wu, Z. Hu, C. Yan, and Y. Yang, "Improving person re-identification by attribute and identity learning," *Pattern Recognition*, 2019. 17, 18

[14] M. Naphade, Z. Tang, M.-C. Chang, D. C. Anastasiu, A. Sharma, R. Chellappa, S. Wang, P. Chakraborty, T. Huang, J.-N. Hwang, and S. Lyu, "The 2019 ai city challenge," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019. 19

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016. 21

[16] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 22

[17] R. Yu, Z. Zhou, S. Bai, and X. Bai, "Divide and fuse: A re-ranking approach for person re-identification," *ArXiv*, vol. abs/1708.04169, 2017. 25

[18] Z. Zhong, L. Zheng, D. Cao, and S. Li, "Re-ranking person re-identification with k-reciprocal encoding," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3652–3661, 2017. 26

[19] X. Wang and R. Zhao, *Person Re-identification: System Design and Evaluation Overview*, pp. 351–370. 01 2014. 27

[20] M. Zhu, "Recall, precision and average precision," 2004. 27